# AUSTRALIAN OS9 NEWSLETTER

EDITOR :
Gordon Bentzen
8 Odin Street
SUNNYBANK Qld 4109

(07) 345-5141

JUNE 1989

# AUSTRALIAN OS9 NEWSLETTER
## Newsletter of the National OS9 User Group

EDITOR : Gordon Bentzen

HELPERS : Bob Devries and Don Berrie

SUPPORT : Brisbane OS9 Level 2 User Group.

Well, this makes issue No 12. in what I guess we could call Vol. 2 of the National OS9 Newsletter, and I think that it is now an appropriate time to examine what we have achieved over the past 12 months.

First of all, we have attempted to be an information sharing device. For that to have been successful, we need to have information to share. Indeed, we have had some interesting tales to tell.

Secondly, we have tried to be a vehicle for people to share their creations with other users who have similar interests.

Thirdly, we have attempted to be a unifying force to make sure that our favourite system doesn't disappear into oblivion.

Have we achieved any or all of these? We (Gordon, Bob and Don) have to say that, frankly, we are disappointed! Apart from a few notable exceptions, the feedback of any kind has been sadly lacking. Providing a newsletter each month is not a task to be taken lightly. To produce a publication which averages 11 pages of text (at 115 characters per line) for each issue is a major undertaking, and as we said in the very first issue, we could not continue to do that without the assistance of the readers.

We feel that we have an obligation to the subscribers. But surely, the subscribers also have some obligations to their fellow readers. Time after time, we have requested input from the readers. It doesn't matter at what level the input is, whether is is basic beginners stuff, or material for the most advenced users, the important thing is that it provides ideas.

Don't worry that your ideas may seem trivial (you know that you are interested in ANYTHING that even remotely relates to OS9), or are not written in polished literary style (we can fix that, at least in our own special way). The important thing is that we get the feedback. Hell, this is a communal effort. Let's see if we can make it that.

Having got that off our chests, the good news is that we have decided to continue to produce the newsletter for another twelve months (starting, of course, from September). The subscription will remain the same as last year, that is $18.00. I guess that you could take this as a gentle reminder that you should budget for your newsletter subscription in the next couple of months. That applies to everybody, regardless of when you actually subscribed. All readers will have recieved back issues to take them to the common renewal date starting in September. (SCREAM IF YOU HAVEN'T)

We are fortunate to have recieved some public domain material from the USA recently. Some of that is presented in this issue. Don has bitten the bullet, and has started into OSK (Atari style), and he has included some initial thoughts for your information. Bob presents the next installment in his database series. We have also included some further articles which we are sure will be of interest to you.

We have again had some problems with people being unable to read disks when returned from us. Once again, please, when sending disks for copies of material from our public domain library, please format your disks in your own drives to your desired format before you send them to us. That way, you will be sure that you can read them when they are returned.

We hope you enjoy our current issue, until next month. Regards. Gordon Bentzen.

## THE OS-K EXPERIENCE

Well, there comes a time when we all have to bite the bullet. One day recently, I was using my MS-DOS machine (as a terminal to my CoCo running OS-9) and I thought, why do I keep this thing, surely not just as a terminal for the CoCo. However, it had become just that, a rather expensive terminal (with a 30 mB hard drive). So, I reasoned, if that is all that I really use it for, why not sell it and get something else? Great idea.

So what should I buy? Obviously a 68000 machine, but!! The great choice! An Atari, or an Amiga. Finally, I found a second-hand Atari 520ST with a 1 mB ram expansion, and price decided the issue for me. (A Macintosh was out of the question - for obvious reasons)

So to ATARI OS-9. Booted the system up, and familiar things started to happen. Drive lights came on, and yes, there is the setime prompt. Mmmm accepts the same familiar input string for the date. Then on the screen: 2, 3, 4, 5-6-7, etc. Heck, what's all this? Suddenly, the background turns from white to blue, and the foreground turns from black to white, and everything stops!! What's happened? Press a key, and suddenly you are prompted to supply a username. Reply with a return, and the screen fills with a message which is obviously the login command's MOTD file. The prompt finally appears - Super:

What do we do first? Of course, dir /D0. That works, let's try the execution directory. Try dir x. What? Error #216. Try dir /D0/CMDS. That seems to work fine. Finally (didn't read the manual of course) try dir -? and, wow!! Inbuilt help. What, dir -x, try that. Yes that's the trick ... it works fine. Wildcard support? Of course. Just like UNIX. But hang on, what were all those numbers that appeared when the system was booting up. I know, I will have a look at the startup file. List /D0/startup. This is what I got:

```
-t -np


      ********************************************************
      *             OS-9/68000 Level I  V1.2                 *
      *   Copyright 1985 by Microware Systems Corporation    *
      *                                                      *
      *                 ATARI 1040 STF                       *
      *                                                      *
      ********************************************************
      *                                                      *
      *                 initializing ...                     *
      *                                                      *
      ********************************************************


-nt
*
* initialize the system
*
setime -s -d                ;* start system clock
link shell cio link         ;* make shell, cio and link stay in memory
iniz d0 d1 r0               ;* initialize devices
xmode /t1 baud=19200        ;* terminal at 19200 baud
iniz t1                     ;* initialize serial port
kill 2
kill 3
skey /d0/sys/scred.sky &    ;* load keyboard image for scred
chd /r0                     ;* change to ramdisk
makdir SYS                  ;* make SYS-Directory
copy /d0/SYS/* -w=SYS -pb=20 &;* and copy SYS-Files
xmode /term pause
```

```
load load mdir dir scred &    ;* load some useful utilities
load ident copy procs mfree &
load date deldir del makdir &
load attr build list login &
tsmon /t1 &                   ;* activate user terminal
setime </term                 ;* read current time and date
ex tsmon /term
```

And as it turns out, all of those numbers are process numbers, all started from the startup file, and running concurrently. And that's the real thing that takes getting used to. The speed of the system, (especially after being used to CoCo Os-9) and OS-K's ability to run many concurrent processes without slowing appreciably. It is perfectly feasible to be DSAVEing from one disk to another, listing a file to the printer, and using the text editor, all at the same time. You can do it with a CoCo provided that you have a "no-halt" controller, and can accept the slow keyboard response, but isn't anywhere near as elegant as with OS-K.

So this is just a starter, I will continue with my explorations of OS-K, and keep you informed over the next few months.

Don Berrie.

oooooooooo0000000000oooooooooo

## The OS9 Operating System

I will continue this month with more comments on the OS9 operating system for the CoCo Color Computer which will be based on OS9 Level 2 with some differences noted as applicable to Level 1.

OS9 Level 2 Modules. Some modules have been discussed in earlier newsletter articles, so let's continue with a description of the function of a few more.

| | |
|---|---|
| INIT | System initialization table |
| CLOCK | Software routine time management |
| IOMAN | Input/output management |
| RBF | Random block file manager |
| SCF | Sequential character file management |
| PIPEMAN | Pipe file management |
| CC3DISK | Color computer disk driver |
| CC3IO | Color Computer input/output driver |
| VDGINT | Video display generator - low level routines |
| GRFINT | High level graphics code interpreter and interface |
| WINDINT | all the functions of GRFINT plus support of Multiview functions |

The OS9 Operating system can be described as having five (5) levels of operation, each of which require specific module types to perform the required functions.

**First level :** This level contains the modules needed for "bootstrapping" the system. This includes the Kernel, OS9P2 (part two of boot), Clock and INIT.

**Second level :** This level contains the Input/Output manager IOMAN which is required for performing all I/O processing supported by OS9.

**Third level :** This level contains the file managers which perform I/O request processing for similar classes of I/O devices.

**Fourth level :** This level contains the device drivers to handle I/O functions for specific I/O controller hardware.

**Fifth level :** This system's fifth level contains the device descriptors which are actually small tables that define the device logical name, device driver and file manager for each I/O port.

I have covered the Kernel and Boot modules in an earlier article, so let's move on to the I/O function and modules.

The I/O system provides all I/O services for user programmes and OS9 itself. All I/O system calls are received by the kernel and passed to the I/O manager for processing. The I/O manager performs processing, such as the allocation of data structures for the I/O path and calls the file manager, device driver and device descriptor to do most of the work.

The I/O manager (**IOMAN**) as well as providing the first level service of I/O system calls and passing data to the appropriate file manager, also maintains two important OS9 data structures; the DEVICE TABLE and the PATH TABLE.

When a path is opened, the I/O manager tries to link to a memory module which is named in the pathlist. This module will be a **Device Descriptor** which contains the names of the device driver and file manager for that device.

**RBF**, **SCF** and **PIPEMAN** are the standard re-entrant file managers, each of which processes the raw data stream to or from a class of device drivers that have similar operational characteristics. A file manager removes as many unique characters as possible from I/O operations to ensure that similar devices conform to the OS9 standard I/O and file structure. The file manager is responsible for mass storage allocation and directory processing if these are applicable to the class of device it serves. File managers usually buffer the data stream and issue requests to the kernel for dynamic allocation of buffer memory.

They can also monitor and process the data stream, e.g. adding line-feed characters after carriage-return characters. OS9 can have a number of **file manager** modules.

**CC3Disk** (level 2 - CoCo3) and **CCDisk** (level 1) are the system's device drivers (fourth level) which of course are the drivers for the disk drive hardware. The device driver modules are subroutine packages that perform basic, low level I/O transfers to or from a specific type of I/O device hardware controller.

OS9 systems can include many other device drivers to handle I/O functions for specific hardware types. e.g. ACIAPak and CC3Hdisk could be the device drivers to handle specific hardware types if the system includes such hardware.

**CC3IO** is a standard device driver which handles Term_VDG, and Term_Win in a CoCo level 2 system. **CCIO** is the video/keyboard driver used on a level 1 system. CC3IO also utilizes interface modules VDGInt, GRFInt or WindInt depending on the device descriptor being used.

The **VDGInt** (VDG interface) performs both interface and low level routines for VDG CoCo 2 compatibility and has limited support for high-res screen allocation. The **GRFInt** interface provides the standard code interpretations and interface functions. The **WindInt** interface comes with the Multi-View package and provides all the functionality of the standard GRFInt plus support of Multi-View features. So if WindInt is used, exclude GRFInt. Term_VDG uses CC3IO and VDGInt. Term_Win and all window descriptors W, W1, W2, etc. use CC3IO plus GRFDrv, GRFInt WindInt as required.

The system's fifth level, **Device Descriptors**, include those mentioned above Term_VDG, Term_Win, W, W1, etc. and also the RBF (random block file) type devices e.g. D0, D1, R0, H0 etc. Device descriptor modules are small, non-executable modules each of which provides information that associates a specific I/O device. Each such module contains its own logical name, device driver and file manager names together with initialization parameters.

Unlike the device drivers and file managers, which operate on classes of devices, each device descriptor is tailored to a specific device. Every device on the system must therefore have a device descriptor.

Well I have just about run out of steam for this month, so I hope that the above is useful to at least some readers. The OS9 Technical manuals for Level 1 and Level 2 will provide more details on the functions and format of the different module types if you need it.

This month's "prattling" has reminded me of some of the diffulties in setting up device drivers and descriptors to suit particular hardware. A good example is that associated with disk drives. Microware's OS9 Level 2 for the

CoCo 3 comes with many disk drive descriptors for almost any requirement. One notable omission is a descriptor for an 80 track double sided D0. If you need to make changes to the Level 2 descriptors, refer to the article by Bob Devries in our Jan-Feb '89 newsletter.

It seems that some OS9 Level 1 users are also restricted by the lack of suitable disk drive 'drivers' and 'descriptors'. OS9 will allow the use of a 40 track double sided drive as a 40 cylinder device. This results in a 40 track double sided drive using both sides as if it were one device, 1440 sectors. The problem is that the Level 1 driver CCDisk is hard coded for a single sided drive and does not look at the descriptor for this information in the same way the Level 2 CC3Disk does. The way I got around this problem on Level 1 was to replace CCDisk with the D.P.Johnson SDisk. SDisk supports all possible combinations as well as non-Tandy OS9 disk formats. The only problem is that SDisk is NOT public domain, so I can not help you with that.

This may be a project for a future newsletter, or perhaps someone out there already has a patch for the CCDisk module. Can anybody help?

Well that's all for this month.

Regards, Gordon

oooooooooo0000000000ooooooooo

**On how to convert those two-disk programmes to 40 or 80 track format.**
**By Bob Devries.**

From WA comes a request for a discussion on how to transfer those programmes like DeskMate and Home Publisher, which come on two disks, or on a flippie, onto a 40 or 80 track disk, and still have them bootable.

Well, here's how I would suggest you do it. First, format a disk to the format you usually use, i.e. 40 tracks double sided or whatever. Put this in a cool dry place for future use....

Now, either use the RAMDISK or create a new working directory (it must not have any files in it) and make it your working directory (CHD /R0). for the rest of this to go ahead smoothly, you must have the following programmes:

LS or FLS or FILES or D

(a programme which lists files from the current directory to STDOUT so it can be PIPED), and

MODBUSTER or BOOTSPLIT

(a programme which can split merged files like OS9BOOT into separate files).

SAVE

(the save programme from level one OS9 or from the development system disk may be used.)

(LS is available on the Home Publisher disk in the CMDS directory)
(D is available in our Public Domain library in volume 21)
(BOOTSPLIT is available on Public Domain library volume 0)
( or see my re-hashed version elsewhere in this issue)
(MODBUSTER is supplied with SDisk3 or SDisk from D.P.Johnson)

Put the DeskMate disk (or whichever other you're doing) into drive /D1 and type:

MODBUSTER /d1/OS9BOOT or use BOOTSPLIT (a bit more tedious)

This will split the OS9BOOT file into its separate components so we can operate on them. DELete the disk device descriptors D0, D1, and if it's there, DD. Now use the SAVE command to save those same descriptors from memory to your current directory like this:

```
SAVE D0
SAVE D1
SAVE DD
```

Remove the original DeskMate Disk, and find that freshly formatted disk and insert it into drive /D1. Now where did I put that thing ? Now we can create the new boot on that disk with this command:

```
LS ! OS9GEN /D1
```

You may use D or FLS instead of LS. What happens here is that LS lists the filenames in the current directory to STDOUT which we PIPE into the input of OS9GEN which in turn uses these names to create the new bootfile on the disk in /D1.

When the above command is finished, you'll have a disk with a bootfile on it with the correct disk descriptors in it. What we now need to do is copy across all the files. Use DSAVE or DIRCOPY to copy the files of both disks onto the one new disk. You'll very likely get some errors like 218 (file already exists) because some of the files will be duplicated on the two disks.

When you've finished all that, you'll have a bootable 40 or 80 track version of DeskMate or Home Publisher or whatever.

oooooooooo0000000000ooooooooo

**A Database in C.**

**By Bob Devries**

Here is part four of my on going series of the address book database in the C programming language. This section is the delete() function which blanks out and removes the currently displayed record from the database. It also calls the save() function (coming soon...).

The delete function does the very simple task of asking if you're sure you want to go ahead, and when you answer 'Y' it sets all the fields to NULL ('\0') and sets the REPLACE variable to TRUE and calls save() to store it on the disk. It passes the record number and the replace variable to save(). If you type 'N' it erases the question, and goes back to the command line of the programme.

Well, here's the next exciting chapter. See you again soon....

```
delete(recno)
int recno;
/* delete() sets the first char of the char strings in the structure to */
/* null and the postcode to 0 then calls save() with replace set to TRUE */
/* so that it will place it over the top of the current record */
{
        int replace;
        char ch;

        replace = TRUE;                 /* replace current record */
        cursor(10,23);
        eraselin();                     /* erase message line */
        cursor(10,23);
        printf("Are you sure ? ");
        while ((ch != 'Y') && (ch != 'N'))
                {
                ch = toupper(getch());
                switch (ch)
                        {
                        case 'Y':
                                mail.surname[0] = '\0';
```

```
                        mail.firstname[0] = '\0';
                        mail.street[0] = '\0';
                        mail.city[0] = '\0';
                        mail.state[0] = '\0';
                        mail.postcode = 0;
                        recno = save(recno,replace);
                        cursor(10,23);
                        eraselin();      /* erase message line */
                        break;
                case 'N':
                        cursor(10,23);
                        eraselin();
                        return;
                }
        }
}
```

oooooooooo0000000000ooooooooo

**A modification for Karl Kreider's Bootsplit.**
**By Bob Devries**

I took pity on all of you people out there who are not able to use modbuster, and made some modifications to the BASIC09 programme so that it will split all the modules in the merged file you tell it. Only one gotcha, you must have IDENT in your current EXECUTION directory and of course RUNB to run the packed version of this programme. The programme creates a temporary file called 'temp' in the current DATA directory, but deletes it before it quits. Have fun. Oh by the way, it is sssllllloooowwwww.

RDV

```
PROCEDURE emerge
0000
0001      (* boot file splitter *)
0019
001A      (* after H.Snyder *)
002E      (* from the public domain *)
004A      (* by Karl Kreider with mods by Bob Devries *)
0078      (* was once called 'BootSplit' *)
0099
009A      DIM mhead,msize,mofs,inpath,outpath,rpath,pts:INTEGER
00B9      DIM index:REAL
00C0      DIM name,mname,infile,rname:STRING
00D3      DIM char:BYTE
00DA      DIM found:BOOLEAN
00E1
00E2      INPUT "file to break up - ",infile
00FD      SHELL "ident "+infile+" -s >temp"
0117      OPEN #inpath,infile
0121      OPEN #rpath,"temp":READ
0130      LOOP
0132      EXITIF EOF(#rpath) THEN
013C      ENDEXIT
0140        INPUT #rpath,rname
014A        pts=SUBSTR(".",rname)
0156        name=RIGHT$(rname,LEN(rname)-pts-1)
016A        name=LEFT$(name,LEN(name)-1)
017A        PRINT name
```

```
017F        GOSUB 1000
0183        IF found THEN
018C          CREATE #outpath,name:WRITE
0198          SEEK #inpath,index
01A2          FOR j=1 TO msize
01B5            GET #inpath,char
01BF            PUT #outpath,char
01C9          NEXT j
01D4          CLOSE #outpath
01DA        ELSE PRINT name; " not found"
01EF        ENDIF
01F1      ENDLOOP
01F5      CLOSE #rpath
01FB      SHELL "del temp"
0207      CLOSE #inpath
020D      END
020F
0210 1000 (* module finder *)
0226      found=TRUE
022C      index=0
0234      msize=0
023B      REPEAT
023D        index=index+msize
024A        SEEK #inpath,index
0254      EXITIF EOF(#inpath) THEN found=FALSE
0263      ENDEXIT
0267        GET #inpath,mhead
0271        GET #inpath,msize
027B        GET #inpath,mofs
0285        SEEK #inpath,index+mofs
0294        mname=""
029B        REPEAT
029D          GET #inpath,char
02A7          mname=mname+CHR$(LAND(char,$7F))
02B8        UNTIL char>127
02C3      UNTIL name=mname
02CF      RETURN
```

**NATIONAL OS9 USER GROUP APPLICATION / RENEWAL FORM**
**********

[ ] **SUBSCRIPTION RENEWAL** [ ] **NEW SUBSCRIPTION**

Surname : _____ First Name :_____ Title (Mr.,etc): ____

Street : _____

Suburb : _____ State :_____ Postcode : _____

Home Phone :_____ Business Phone : _____

Do you run OS9 Level 1 [__] and/or OS9 Level 2 [__] (please tick)

Type of Computer for OS9 :_____

Diskette Size (inches): ____ Number of Tracks: ____ Sides:____ Number of Drives :__

Special Interests : _____

Can you contribute articles to this Newsletter ? _____

Date : ____/____/____ Signature :_____

Amount Enclosed: $ _____ ( $18-00 will cover you for 12 months)
**(cheques payable to National OS9 User Group)**

**NOTE:** All current subscriptions Expire AUGUST 31st, 1989 (Two Year subscribers AUG 1990)

- Renewals Commence SEPTEMBER 1989 (SEPT 1990)

All current subscribers should now have back-copies from SEPTEMBER 1988 if you are missing any editions please note here the missing editions.

[ ] SEP [ ] OCT [ ] NOV [ ] DEC [ ] JAN/FEB [ ] MAR [ ] APR [ ] MAY [ ] JUN [ ] JUL [ ] AUG

Please Return Completed Form to :-

**NATIONAL OS9 USER GROUP**
**C/o Gordon BENTZEN**
**8 ODIN STREET,**
**SUNNYBANK QLD 4109**